REMARKS

This responds to the Office Action mailed on September 18, 2008.

Claims 1, 3, 4, 7, 8, 10, 11, 14, 16, 20, 21, 22, and 23 are amended and claims 2, 6, 9, and 13 are canceled. With no claims being added, claims 1, 3-5, 7, 8, 10-12, and 14-23 are now pending in this application.

Claim Objections

Claims 21 and 22 are objected to because of the informalities.

Applicant has amended claims 21 and 22 to overcome the objections. Therefore, Applicant respectfully requests to withdraw the objections.

Claim 23 was objected under 37 CFR 1.75(c).

Applicant has amended claim 23 to overcome the objections. Therefore, Applicant respectfully requests to withdraw the objections.

§101 Rejection of the Claims

Claims 8-14, 16, 17, 19, and 20 were rejected under 35 USC § 101.

Applicant has amended independent claims 8, 16, and 20 to overcome the rejections. Support for this can be found in page 17, lines 14-21 of the specification. Therefore, Applicant respectfully requests withdrawal of the rejection and allowance of claims. Claims 10, 14, 19, and 17 are also allowable at least because they each depend directly or indirectly from a respective one of amended independent claims 8, 16, and 20, all of which are allowable as explained above.

Claim 21, 22 and 23 was rejected under 35 USC § 101.

Applicant has amended independent claim 21 to overcome the rejection. Support for the amendment can be found in page 17, lines 8-9 of the specification. Therefore, Applicant

respectfully requests withdrawal of the rejection and allowance of the claim. Claims 22 and 23 are also allowable at least because they each depend directly or indirectly from amended independent claim 21, all of which are allowable as explained above.

§103 Rejection of the Claims

Claims 1-4, 7-11, and 14-19 were rejected under 35 USC § 103(a) over Hughes et al. (hereafter Hughes) (US Pat. 6,519,768) in view of Bharadwaj (US Pat. 5,894,576).

Huhges describes "source code instructions are **translated into target code instructions for execution** on **a particular processor**". In contrast, amended independent claims 1, 8, and 15 recite "**generating code for scheduling the execution of binary code translated from a source format to a target format**". Further, in contrast, amended independent claim 16 recites "binary code **translator that can operate between any processor or platform types** for translating binary code from a source format to a target format for execution on a target processor".

Hughes, in col. 2, lines 5-8, describes "**for each instruction in an input block of source code instructions, selecting an appropriate template** for that source code instruction and **appending this template to an output block of target code instructions**". In contrast, amended independent claims 1, 8, and 15 recite **identifying a set of target instructions semantically equivalent to a given source instruction**, and wherein the **set of target instructions is identified in a translation template** associated with a given source instruction, said **template being a component of a translator program for translating instructions** in the source format into instructions in the target format. Further, in contrast, amended independent claim 16 recites "a set of translation templates, **each template arranged for providing a set of target format instructions which together are semantically equivalent to an associated source format instruction**, wherein each translation template is associated with a given source instruction, said **template being a component of a translator program for translating binary code in the source format into the target format**".

Hughes, in col. 3, lines 57-65, describes "For **each marker value in the template, the Initialise Templates process inserts an entry (referred to herein as a "fix-up" entry) in the**

data structures 16. The **fix-up entry identifies the location of the marker value** and specifies the data type of the constant value that is to be inserted into the code at translate time (run time) to replace the marker value. Similarly, for each call in the template, the Initialise Templates process inserts a fix-up entry in the data structures **16**, identifying the location of the call". In contrast, amended independent claims 1, 8, and 15 recite **assigning an identifier to one or more of said target instructions for use by a code analyser in scheduling the processing of said set of target instructions in accordance with the identified data dependencies**, and wherein **data dependencies are represented by a directed acyclic graph**, and the identifying step identifies said dependency signaling an appropriate edge in the set of target instructions to said code analyser.

Hughes, in col. 3, lines 65 and col. 4, line 5, does not disclose "**a set of data transformation routines arranged to transform data from a source format instruction into the appropriate parts of each target format instruction** provided by the corresponding translation template" as recited in amended independent claim 16.

Bharadwaj describes "**method of instruction scheduling in superscalar or very large instruction word (VLIW) processors that reduces the harmful effects of compensation code**". Further, Bharadwaj, in col. 1, lines 13-35, describes "The final step in the compiler's translation of the source code program is code generation, during which the compiler creates the final representation of the source code program in the target language. One step that is performed during the code generation process is called instruction scheduling. During instruction scheduling, individual instructions of the target code (or instructions of an intermediate representation of the target code) are rescheduled so that more instructions execute in each clock cycle, enabling faster execution of the target code instructions and more efficient use of machine resources". Furthermore, Bharadwaj, in col. 6, lines 13-35, describes "if instructions were organized in block C 104 so as to fully use the machine resources at each clock cycle, nothing else could be scheduled into that block, and therefore **compensation block 109 would not be very important and could be marked dummy**. However, if instructions in block C 104 were such that only a portion of the machine resources were being utilized in a given clock cycle, it would be desirable to schedule other instructions into that block to fully use the machine resources. In that instance, compensation block 109 would be necessary to accept compensation

copies of instructions that are scheduled into block C 104. Under another method, a **compensation block would be marked dummy if it sat along a very high probability control flow path**", "In another embodiment, **compensation blocks might not be inserted. Such an embodiment would be the functional equivalent of inserting the compensation blocks, as in this embodiment, and designating them all as dummy blocks**", and "the **control flow analyzer 66 calculates various control flow data. (In computing all such data, edges not in the region are ignored.) For example, it computes reachability, i.e., for each block in the control flow diagram**, it determines which blocks are reachable from it".

In contrast, amended independent claims 1, 8, and 15 recite **identifying data dependencies in said target instructions by analyzing the set of target instructions** and assigning an identifier to one or more of said target instructions for use by a code analyser in **scheduling the processing of said set of target instructions in accordance with the identified data dependencies**, and wherein **data dependencies are represented by a directed acyclic graph**, and the identifying step identifies said dependency signaling an appropriate edge in the set of target instructions to said code analyser. This type of **data dependencies represented in a directed acyclic graph** and **identifier to one or more of said target instructions for use by a code analyser in scheduling the processing of said set of target instructions in accordance with the identified data dependencies** is not disclosed in Bharadwaj.

Further, in contrast, amended independent claim 16 recites "**a set of data transformation routines arranged to transform data from a source format instruction into the appropriate parts of each target format instruction** provided by the corresponding translation template" and "**a set of analysis routines arranged to identify data dependencies** in a template for causing generation of data for use by a code scheduler in scheduling the execution of translated code on said target processor, and wherein **data dependencies are represented by a directed acyclic graph**, and the identifying step identifies said dependency signaling an appropriate edge in the set of target instructions to said code analyser".

Applicant respectfully asserts that Hughes and Bharadwaj references fail to support a *prima facie* case of obviousness because, the cited references fail to teach or suggest all of the elements of the Applicant's invention, such as **generating code for scheduling the execution of binary code translated from a source format to a target format, identifying a set of target**

instructions semantically equivalent to a given source instruction, and wherein the **set of target instructions is identified in a translation template** associated with a given source instruction, said **template being a component of a translator program for translating instructions** in the source format into instructions in the target format, **identifying data dependencies in said target instructions by analyzing the set of target instructions** and assigning an identifier to one or more of said target instructions for use by a code analyser in **scheduling the processing of said set of target instructions in accordance with the identified data dependencies**, and wherein **data dependencies are represented by a directed acyclic graph**, and the identifying step identifies said dependency signaling an appropriate edge in the set of target instructions to said code analyser.

Further, Applicant respectfully asserts that Hughes and Bharadwaj references fail to support a *prima facie* case of obviousness because, the cited references fail to teach or suggest all of the elements of the Applicant's invention, such as "A binary code translator that operates between any processor or platform types for translating binary code from a source format to a target format for execution on a target processor, the source format differing from the target format", "a set of translation templates, each template arranged for providing a set of target format instructions which together are semantically equivalent to an associated source format instruction, wherein each translation template is associated with a given source instruction, said template being a component of a translator program for translating binary code in the source format into the target format", "a set of data transformation routines arranged to transform data from a source format instruction into the appropriate parts of each target format instruction provided by the corresponding translation template", and "a set of analysis routines arranged to identify data dependencies in a template for causing generation of data for use by a code scheduler in scheduling the execution of translated code on said target processor, and wherein data dependencies are represented by a directed acyclic graph, and the identifying step identifies said dependency signaling an appropriate edge in the set of target instructions to said code analyser".

For the above reasons, amended independent claims 1, 8, 15, and 16 should be found allowable over Hughes and Bharadwaj references, and such action is respectfully requested.

Claims 3, 4, 7, 10, 11, 14, and 17-19 depend directly or indirectly from a respective one of amended independent claims 1, 8, 15, and 16, all of which are patentable for the reasons presented above.

For at least the reasons presented above, applicant respectfully requests that the 35 USC § 103(a) rejection of claims 1, 3, 4, 7, 8, 10, 11, and 14-19 be withdrawn.

Claims 5 and 12 were rejected under 35 USC § 103(a) over Hughes in view of Bharadwaj as applied to claims 1 and 8 above, and further in view of Mochizuki (US Pat. 6,016,396).

Huhges describes "source code instructions are **translated into target code instructions for execution** on **a particular processor**". In contrast, amended independent claims 1 and 8 recite "**generating code for scheduling the execution of binary code translated from a source format to a target format**".

Hughes, in col. 2, lines 5-8, describes "**for each instruction in an input block of source code instructions, selecting an appropriate template** for that source code instruction and **appending this template to an output block of target code instructions**". Further, Hughes, in col. 3, lines 57-65, describes "For **each marker value in the template, the Initialise Templates process inserts an entry (referred to herein as a "fix-up" entry) in the data structures 16** . The **fix-up entry identifies the location of the marker value** and specifies the data type of the constant value that is to be inserted into the code at translate time (run time) to replace the marker value. Similarly, for each call in the template, the Initialise Templates process inserts a fix-up entry in the data structures **16** , identifying the location of the call".

Bharadwaj describes "**method of instruction scheduling in superscalar or very large instruction word (VLIW) processors that reduces the harmful effects of compensation code**". Further, Bharadwaj, in col. 1, lines 13-35, describes "The final step in the compiler's translation of the source code program is code generation, during which the compiler creates the final representation of the source code program in the target language. One step that is performed during the code generation process is called instruction scheduling. During instruction scheduling, individual instructions of the target code (or instructions of an intermediate

representation of the target code) are rescheduled so that more instructions execute in each clock cycle, enabling faster execution of the target code instructions and more efficient use of machine resources". Furthermore, Bharadwaj, in col. 6, lines 13-35, describes "if instructions were organized in block C 104 so as to fully use the machine resources at each clock cycle, nothing else could be scheduled into that block, and therefore **compensation block 109 would not be very important and could be marked dummy**. However, if instructions in block C 104 were such that only a portion of the machine resources were being utilized in a given clock cycle, it would be desirable to schedule other instructions into that block to fully use the machine resources. In that instance, compensation block 109 would be necessary to accept compensation copies of instructions that are scheduled into block C 104. Under another method, a **compensation block would be marked dummy if it sat along a very high probability control flow path**", "In another embodiment, **compensation blocks might not be inserted. Such an embodiment would be the functional equivalent of inserting the compensation blocks, as in this embodiment, and designating them all as dummy blocks**", and "the **control flow analyzer 66 calculates various control flow data. (In computing all such data, edges not in the region are ignored.) For example, it computes reachability, i.e., for each block in the control flow diagram**, it determines which blocks are reachable from it".

Mochizuki, in col. 1, lines 7-12, describes "parallel code conversion processing method and system for transferring information of a given content at high speed between computer systems having different code schemes".

In contrast, amended independent claims 1 and 8 recite **generating code for scheduling the execution of binary code translated from a source format to a target format, identifying a set of target instructions semantically equivalent to a given source instruction**, and wherein the **set of target instructions is identified in a translation template** associated with a given source instruction, said **template being a component of a translator program for translating instructions** in the source format into instructions in the target format, **identifying data dependencies in said target instructions by analyzing the set of target instructions** and assigning an identifier to one or more of said target instructions for use by a code analyser in **scheduling the processing of said set of target instructions in accordance with the identified data dependencies**, and wherein **data dependencies are represented by a directed acyclic**

graph, and the identifying step identifies said dependency signaling an appropriate edge in the set of target instructions to said code analyser.

Applicant respectfully asserts that Hughes, Bharadwaj and Mochizuki references fail to support a *prima facie* case of obviousness because, the cited references fail to teach or suggest all of the elements of the Applicant's invention, such as **generating code for scheduling the execution of binary code translated from a source format to a target format, identifying a set of target instructions semantically equivalent to a given source instruction**, and wherein the **set of target instructions is identified in a translation template** associated with a given source instruction, said **template being a component of a translator program for translating instructions** in the source format into instructions in the target format, **identifying data dependencies in said target instructions by analyzing the set of target instructions** and assigning an identifier to one or more of said target instructions for use by a code analyser in **scheduling the processing of said set of target instructions in accordance with the identified data dependencies**, and wherein **data dependencies are represented by a directed acyclic graph**, and the identifying step identifies said dependency signaling an appropriate edge in the set of target instructions to said code analyser.

For the above reasons, amended independent claims 1 and 8 should be found allowable over Hughes and Bharadwaj references.

Claims 5 and 12 depend directly or indirectly from a respective one of amended independent claims 1 and 8, all of which are patentable for the reasons presented above.

For at least the reasons presented above, applicant respectfully requests that the 35 USC § 103(a) rejection of claims 5 and 12 be withdrawn.

Claims 6 and 13 were rejected under 35 USC § 103(a) over Hughes in view of Bharadwaj as applied to claims 1 and 8 above, and further in view of Rasbold et al (hereafter Rasbold) (US Pat. 5,202,975).

Claims 6 and 13 have been canceled.

Claim 20 was rejected under 35 USC § 103(a) as being unpatentable over Hughes.

Huhges describes "source code instructions are **translated into target code instructions for execution** on **a particular processor**". In contrast, amended independent claim 20 recites "Apparatus **that can operate between any platform types for translating machine instructions in source code into equivalent target instructions of a code of a target platform**".

Hughes, in col. 1, lines 55-58, describes "for each instruction in an input block of source code instructions, selecting an appropriate template for that source code instruction and appending this template to an output block of target code instructions". In contrast, amended independent claim 20 recites "a source of **binary translation templates stored in storage means for mapping instructions in the source code into a set of instructions in the code of the target platform**".

Hughes, in col. 3, lines 42-53, does not disclose "dynamic binary translator arranged to be responsive to the machine instructions" as recited in amended independent claim 20.

Applicant respectfully asserts that Hughes reference fails to support a *prima facie* case of obviousness because, the cited reference fails to teach or suggest all of the elements of the Applicant's invention, such as a "source of binary translation templates stored in storage means for mapping instructions in the source code into a set of instructions in the code of the target platform", "a fill and analysis routine generator stored in storage means arranged to be responsive to the templates for generating fill and analysis routines for identifying fillable positions in a template by parsing the template and for generating code to extract and deposit fields from the machine instructions in source code into a precompiled template", and "a dynamic binary translator arranged to be responsive to the machine instructions".

As described above, if the Examiner is using personal knowledge or is taking Official Notice of the elements of claims 20 which are not found in Hughes patent, Applicant respectfully traverses and requests that the Examiner either provide a reference of references which describe such missing elements pursuant to M.P.E.P. § 2144, or submit an affidavit as required by 37 C.F.R. § 1.104(d) (2).

For the above reasons, amended independent claim 20 should be found allowable over Hughes reference.

For at least the reasons presented above, applicant respectfully requests that the 35 USC § 103(a) rejection of claim 20 be withdrawn.

Allowable Subject Matter

Claims 21-23 would be allowable if rewritten or amended to overcome the rejection(s) under 35 U.S.C. 101 and the claim objections, set forth in this Office action.

Applicant has amended claim 21 to overcome the rejection and objections. Therefore, Applicant respectfully request to allow claims 21. Claims 22 and 23 are also allowable at least because they each depend directly or indirectly from amended independent claim 21, all of which are allowable as explained above.

CONCLUSION

Applicant respectfully submits that the claims 1, 3-5, 7, 8, 10-12, and 14-23 are in condition for allowance and notification to that effect is earnestly requested. The Examiner is invited to telephone Applicant's attorney to facilitate prosecution of this application.

If necessary, please charge any additional fees or credit overpayment to Deposit Account No. 08-2025.

Respectfully submitted,

DIBYAPRAN SANYAL

By his Representatives,

Global IP Services, PLLC,
198 F, 27th Cross, 3rd Block,
Jayanagar, Bangalore 560 011
INDIA

Phone: 603-888-7958

Date February 18, 2009 By _____
Prakash Nama
Reg. No. 44,255